



OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

CT336/CT404 Graphics & Image Processing

**Sem 1 (Autumn) 2024-25,
Dr. Nazre Batool**

4th year B.Sc. (CS&I.T.).
2nd year M.Sc. (Software
Development & Design)
1st year M.Sc. (Biomedical
Engineering)
Visiting students



University
ofGalway.i
e



OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

*~15 minutes for your valuable,
honest feedback!*

External evaluator: Ms. Marilla Keating

Lecture 7: Edge & Feature Detection



OLLSCOIL NA GAILLIMHÉ
UNIVERSITY OF GALWAY

- Recap of Last Lecture(s):
 - Image Enhancement & Pre-processing via
 - ◆ Point & geometric transformations
 - ◆ Filtering (smoothing, differentiating) in Spatial and Frequency Domains
 - Binary Image Processing **Pipeline** using Morphological Techniques
 - ◆ Basic morphological operations (erosion, dilation, opening, closing, thinning, thickening)
 - ◆ Extraction of binary regions (connected components)
 - ◆ Distance Transform & Skeletonization
 - ◆ Properties of Binary Regions for choosing regions of interest
 - ◆ *You will have an assignment task on implementing this complete pipeline!*
- Today – start understanding more complex operations on images:
 - ◆ Multiresolution processing & Image Pyramids
 - ◆ Edges
 - ◆ Feature Detectors & Feature Descriptors
 - ◆ Segmentation (Part 1 of 2)
- ◆ **Acknowledgement:** These lecture slides have been prepared using digital materials available with your Text book: 'Image Processing and Analysis by Stan Birchfield, 1st Edition, Cengage Learning'

Multiresolution Processing



OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

- **Multiresolution:** To analyse the image at multiple resolutions (scales)
- **Image Pyramid:**
 - A 40 X 40 region in the original image will occupy only a 20 X 20 region in the downsampled image, a 10 X 10 region in the twice downsampled image, and so forth.
 - Because each successive image is smaller than its predecessor, stacking the images on top of one another yields the shape of a pyramid.
 - The sequence of images is known as an image pyramid.
- How to create image pyramids? Smooth -> downsample -> smooth -> downsample ->
 - Scale-space analysis
 - Image Pyramids (Gaussian and Laplacian)
- Why is multiresolution processing beneficial?

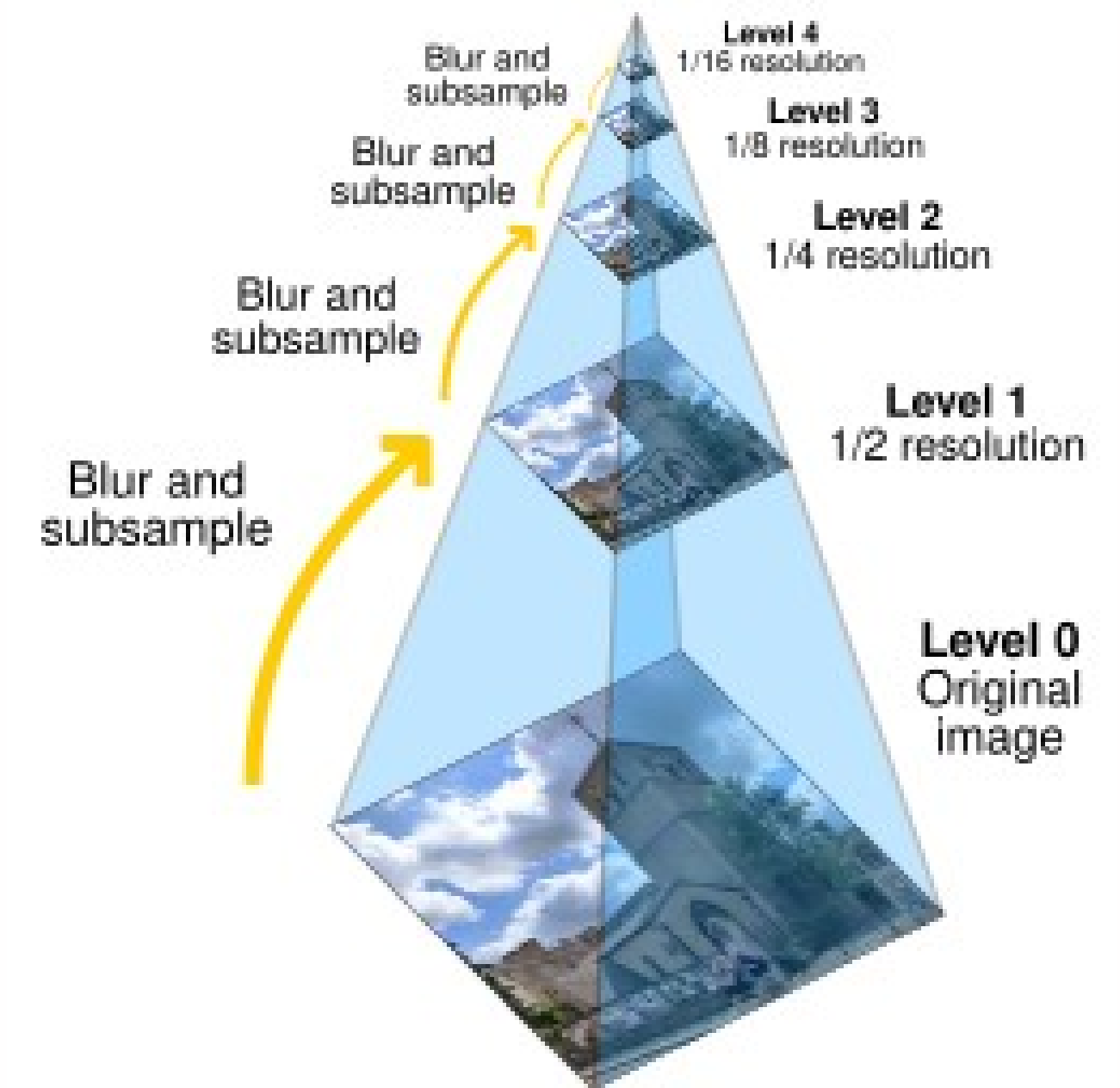


Image source: Wikipedia

Multiresolution Processing

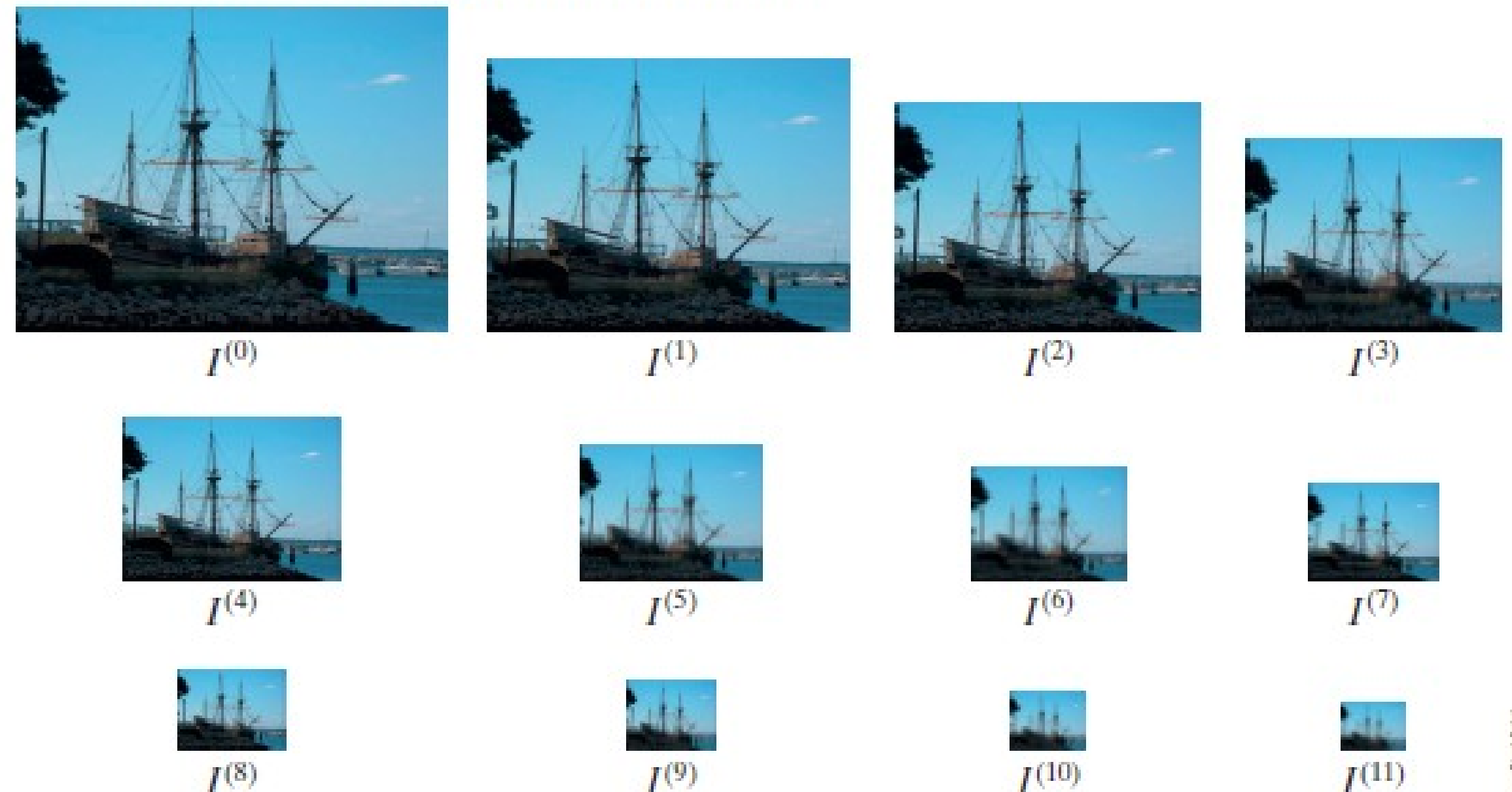


- **Gaussian Pyramid:** smooth an image by convolving with a Gaussian kernel

$$I^{(0)}(x, y) \equiv I(x, y)$$

$$I^{(i+1)}(x, y) \equiv (I^{(i)}(x, y) \otimes \text{Gauss}_{\sigma^2}(x, y)) \downarrow 2$$

Figure 7.2 Twelve levels of a Gaussian pyramid, obtained with $\sigma'^2 = \frac{1}{4}(0.5) = 0.125$ and a downsampling factor of $\sqrt[4]{2}$. Note that $I^{(4)}$ is half as large as $I^{(0)}$ in each direction, and that $I^{(8)}$ is half as large as $I^{(4)}$.



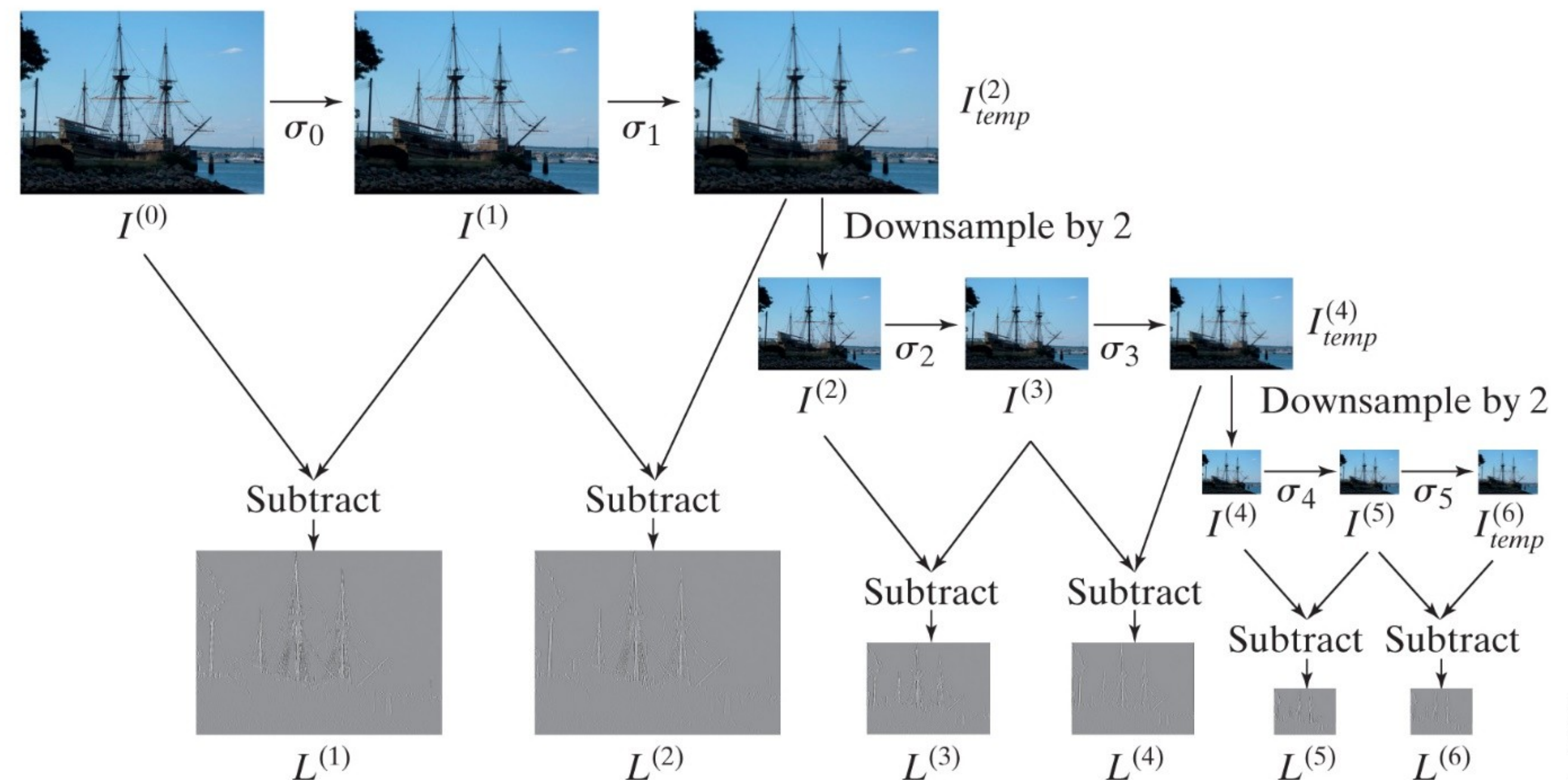
Multiresolution Processing



- **Laplacian Pyramid :**
smooth an image by convolving with several Gaussian kernels of varying variance, then take their difference DoG (Difference of Gaussian) to approximate LoG (Laplacian of Gaussian)

$$L^{(i+1)}(x, y) \equiv (I^{(0)}(x, y) \otimes LoG_{(i+1)\sigma^2}(x, y)) \downarrow (i+1)d$$

Figure 7.3 Laplacian pyramid with $n = 2$ images per octave. The images are successively convolved with a Gaussian, then downsampled at the end of each octave to produce something that closely resembles a Gaussian pyramid. Differences between successive Gaussian-smoothed images yield DoGs, which approximate LoGs. The initial variance is $\frac{1}{2}(0.5) = 0.25$, and the ratio between successive standard deviations is $\rho = \sqrt{2}$.



Multiresolution Processing



OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

- **Scale-space** : To construct a series of images at different scale-space an image is convolved with varying kernel sizes of Gaussian smoothing kernels of varying size and variance ('scale').
- can be seen as an embedding of the original image into a one-parameter family of Gaussian kernels of increasing variance – being the **continuous** parameter.
- The image size remains the same and not down-sampled
- used in further image analysis techniques such as SIFT Features

Figure 7.4 The Gaussian scale space of an image consists of a continuous 3D volume in which each slice is an increasingly blurred version of the original image. Shown here are ten sample images from the scale space.



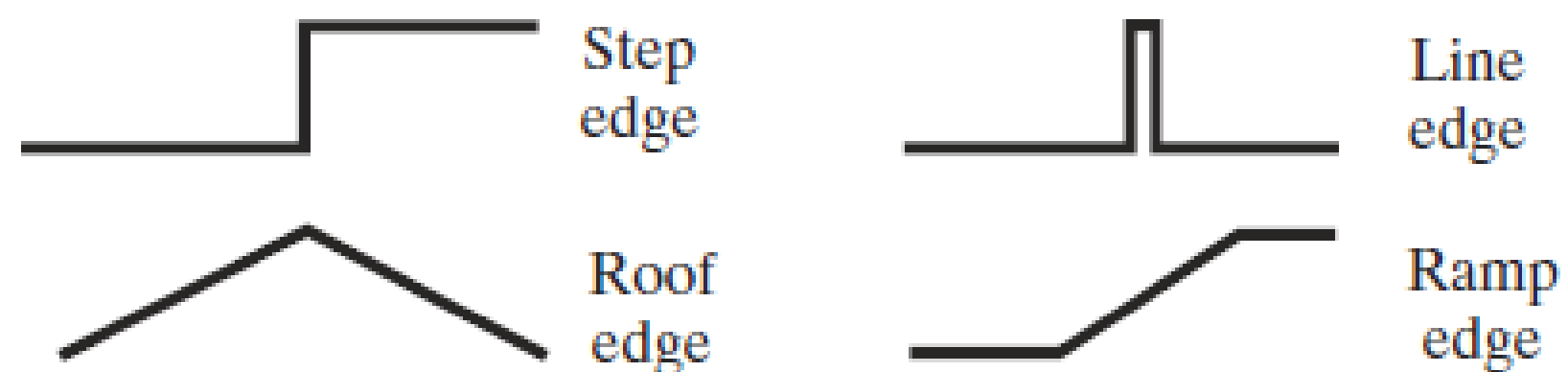
Edge Detection



OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

- **Intensity edges** are pixels in the image where the intensity (or graylevel) function changes rapidly.
 - **Step edge:** occurs when a light region is adjacent to a dark region.
 - **Line edge:** occurs when a thin light (or dark) object, such as a wire, is in front of a dark (or light) background.
 - **Roof edge:** the change is not in the lightness itself but rather the derivative of the lightness.
 - **Ramp edge:** occurs when the lightness changes slowly across a region.
- *Have we done any edge detection before?*

Figure 7.8 Four types of intensity edges.



Edge Detection



OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

Figure 7.6 Intensity edges capture a rich representation of the scene. The scenes and objects in these line drawings are, with little difficulty, recognizable by the average human viewer. From Walther et al. [2011]. For the original images, turn to Figure 7.7.



D. B. Walther, B. Chai, E. Caddigan, D. M. Beck, and L. Fei-Fei, "Simple line drawings suffice for functional MRI decoding of natural scene categories," *Proceedings of the National Academy of Sciences (PNAS)*, 108(23):9661–9666, 2011.

Edge Detection



OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

- **Canny Edge Detector:** a classic algorithm for detecting intensity edges in a grayscale image that relies on the gradient magnitude.
- The algorithm involves three steps:
 - First, the gradient of the image is computed, including the magnitude and phase.
 - Second, in **non-maximum suppression**, any pixel is set to zero whose gradient magnitude is not a local maximum in the direction of the gradient.
 - Finally, **edge linking** is performed to discard pixels without much support.
- Highly regarded for a number of reasons:
 - It has good ability to locate as many edges as possible
 - It is relatively insensitive to noise
 - There will be a minimal distance between its detected edges and the real edges (important when you want to measure/classify extracted objects)
 - It gives only one response to each edge

Edge Detection



OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

- **Canny Edge Detector:** a classic algorithm for detecting intensity edges in a grayscale image that relies on the gradient magnitude & phase.
- The algorithm involves three steps:
 - First, the gradient of the image is computed, including the magnitude and phase.
 - Second, in **non-maximum suppression**, any pixel is set to zero whose gradient magnitude is not a local maximum in the direction of the gradient.
 - Finally, **edge linking** is performed to discard pixels without much support.

- Recap: Gradient Magnitude and **Phase**

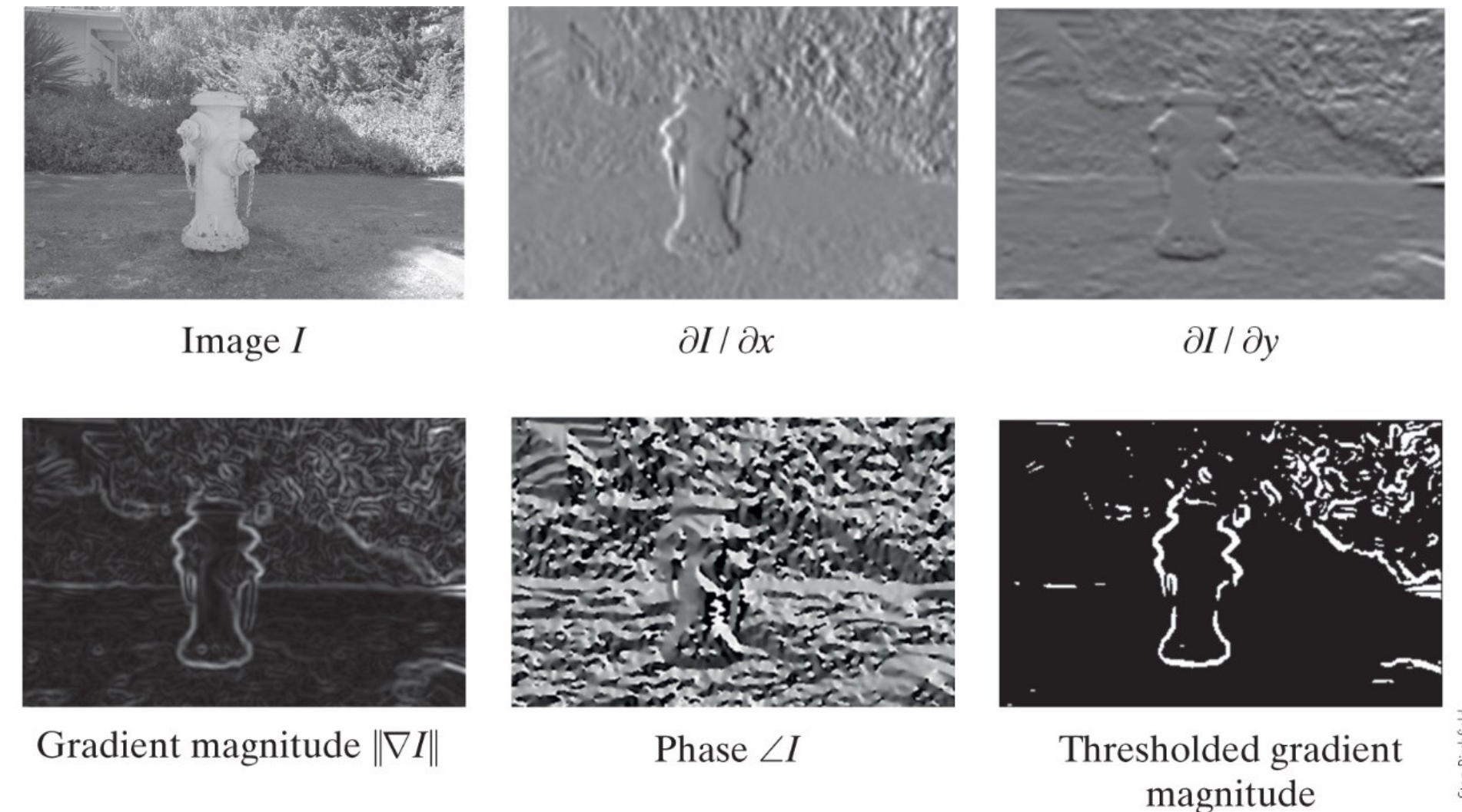


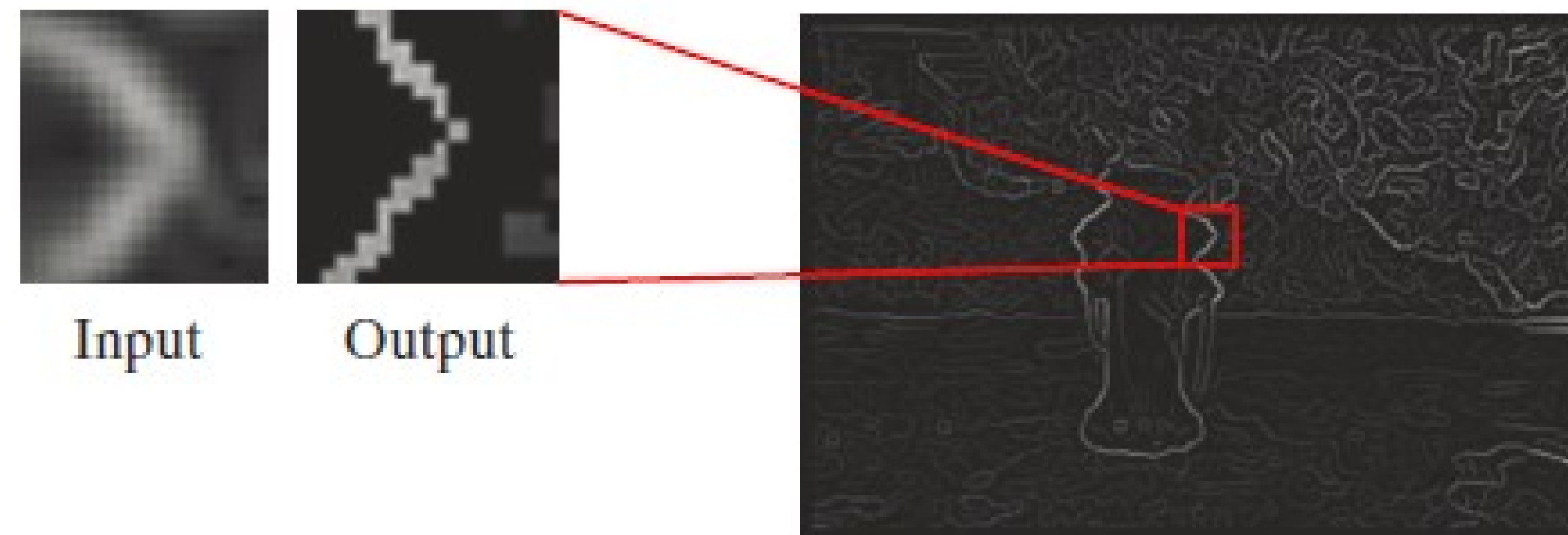
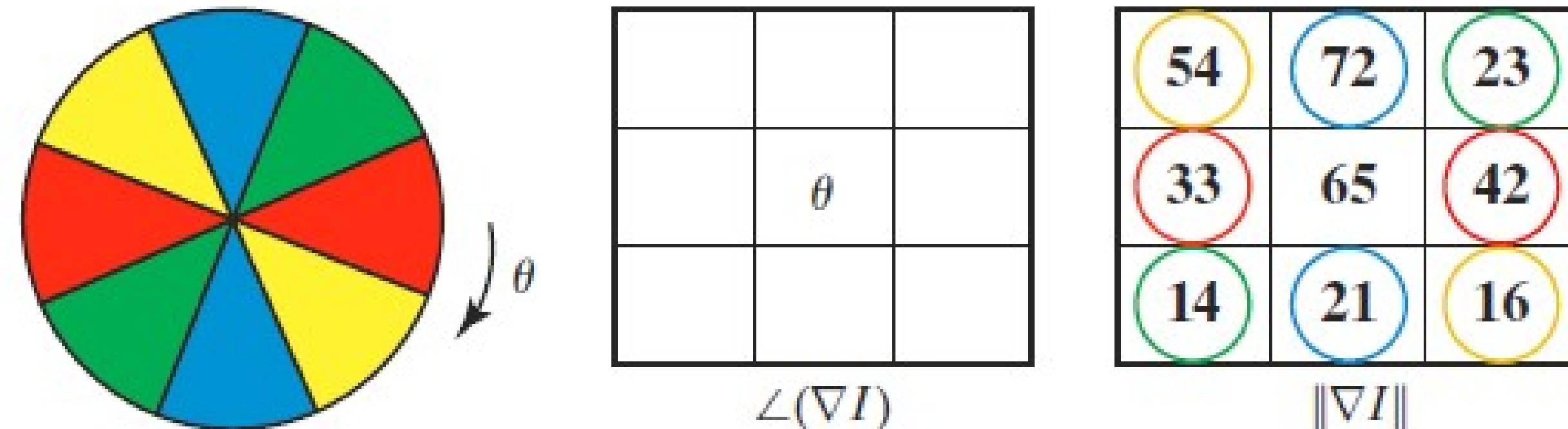
Figure 7.9 Top: An image and its partial derivatives in the x and y directions. Bottom: The gradient magnitude and phase of the image, along with the thresholded gradient magnitude.

Edge Detection



■ Canny Edge Detector: non-maximum suppression

Figure 7.10 Non-maximum suppression. The gradient direction (or phase) θ is quantized into one of four values, shown by the colored wedges of the circle. The quantized phase governs which of the two neighbors to compare with the pixel. If the gradient magnitude of the pixel is not at least as great as both neighbors, then it is set to zero. This has the effect of thinning the edges, as shown in the inset.



Edge Detection



OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

■ Canny Edge Detector: Edge Linking

Figure 7.11 Edge linking with hysteresis, also known as double thresholding or hysteresis thresholding. Thresholding the gradient magnitude with the low threshold produces too many edge pixels (left), while thresholding with the high threshold produces too few edge pixels (middle). Edge linking with hysteresis combines the benefits of both (right), to produce the final Canny edge detector output.

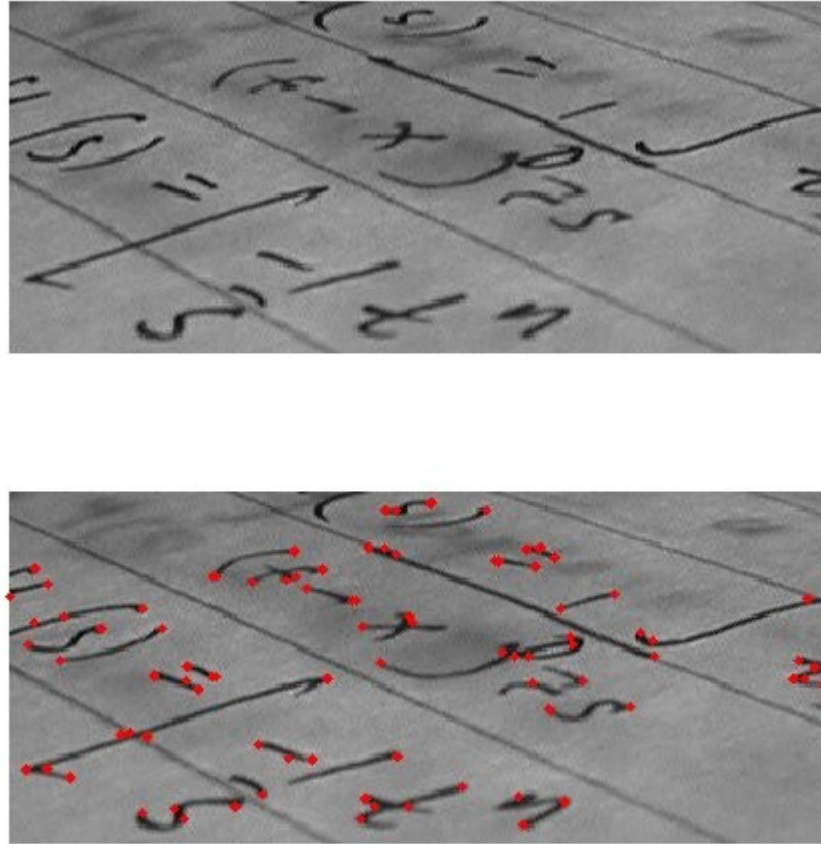


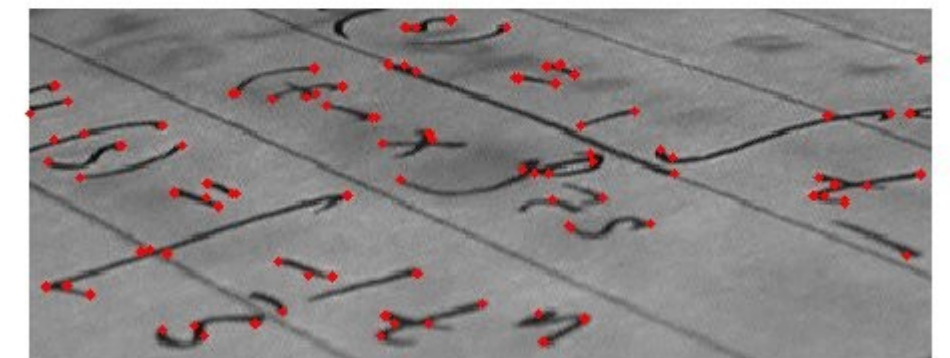
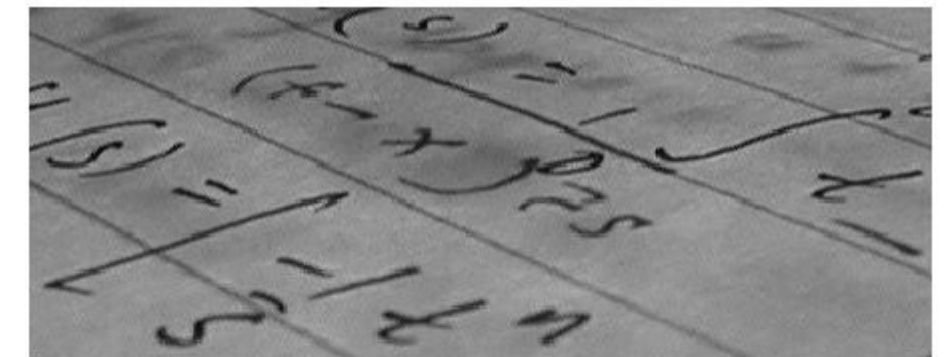
Stan Birchfield

Feature Detection



OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

- An edge detector finds pixels with large gradient magnitude whereas a feature detector finds pixels where grayscale values vary *locally in more than one direction*.
 - One of the common places for features points is at **Corners**.
 - Harris Corner Detector
 - Tomasi-Kanade Feature Detector
 - SIFT Feature Detection
 - *What are detected features useful for?*
 - Frequently used in motion detection, object tracking, image registration, image stitching, 3D modelling, optical flow and visual odometry/SLAM (Robotics).
- 



Feature Detection - Applications



OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

- Applications: Image registration, mosaic, stitching, panorama generation

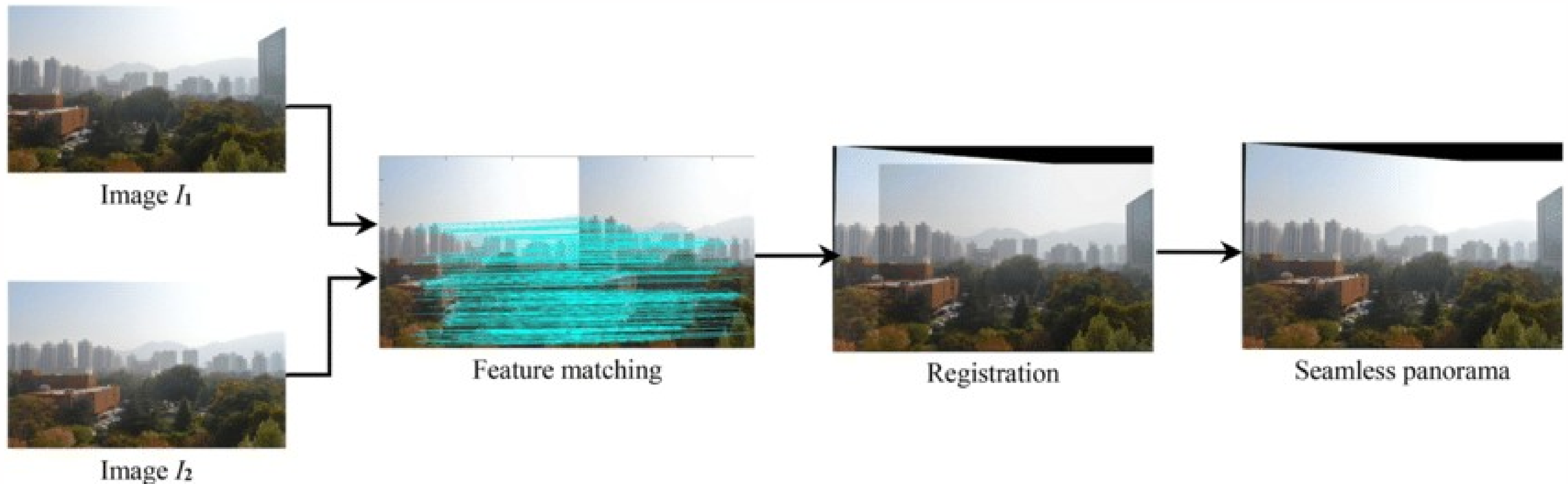


Image source: Wang, Zhaobin & Yang, Zekun. (2020). Review on image-stitching techniques. Multimedia Systems. 26. 10.1007/s00530-020-00651-y.

Feature Detection - Applications

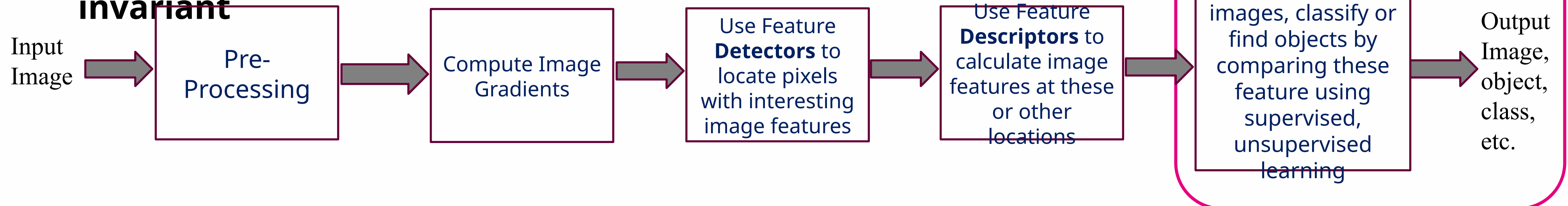


OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

- Applications: Object Tracking and Optical Flow
Nvidia's Pyramidal Lucas-Kanade (LK) Optical Flow algorithm
- SLAM in Robotics using ORB features



- These features are **hand-crafted** (as against those found by deep learning pipelines)
- Good features are **Scale-invariant** and **Rotation invariant**



Feature Detection



- Remember the **Covariance Matrix**, built from central moments, whose **eigen values** were used to find orientation & eccentricity of a (binary) region?

$$\mathbf{C}_{\{2 \times 2\}} = E[(\mathbf{x} - \bar{\mathbf{x}})(\mathbf{x} - \bar{\mathbf{x}})^T \rho(\mathbf{x})] \quad (4.126)$$

$$= \frac{1}{\mu_{00}} \begin{bmatrix} \mu_{20} & \mu_{11} \\ \mu_{11} & \mu_{02} \end{bmatrix} \quad (4.131)$$

- Now we build a **Gradient Covariance Matrix/Autocorrelation Matrix** from image gradients

$$\mathbf{Z} \equiv \sum_{\mathbf{x} \in \mathcal{R}} w(\mathbf{x}) \begin{bmatrix} I_x^2(\mathbf{x}) & I_x(\mathbf{x})I_y(\mathbf{x}) \\ I_x(\mathbf{x})I_y(\mathbf{x}) & I_y^2(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} z_x & z_{xy} \\ z_{xy} & z_y \end{bmatrix} \quad (7.29)$$

- Eigen values of Gradient Covariance Matrix** indicate how the pixel values are varying in different directions.

Feature Detection



- **Eigen values of Gradient Covariance Matrix**

- Both eigen values are large -> pixel values are varying in different directions
- One eigenvalue is large -> indicate an edge with pixel values varying in one direction only.

- **Harris corner detector:** Find pixel values with large 'corneriness' defined as follows

$$\text{corneriness} \equiv \det(\mathbf{Z}) - k (\text{trace}(\mathbf{Z}))^2 \quad (\text{Harris}) \quad (7.30)$$

$$= z_x z_y - z_{xy}^2 - k(z_x + z_y)^2 \quad (7.31)$$

$$= \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2 \quad (7.32)$$

- **Tomasi-Kanade/Shi-Tomasi Detector:** Find pixel values with large 'corneriness' defined as follows

$$\text{corneriness} \equiv \min(\{\lambda_1, \lambda_2\}) = \lambda_2 \quad (\text{Tomasi-Kanade}) \quad (7.33)$$

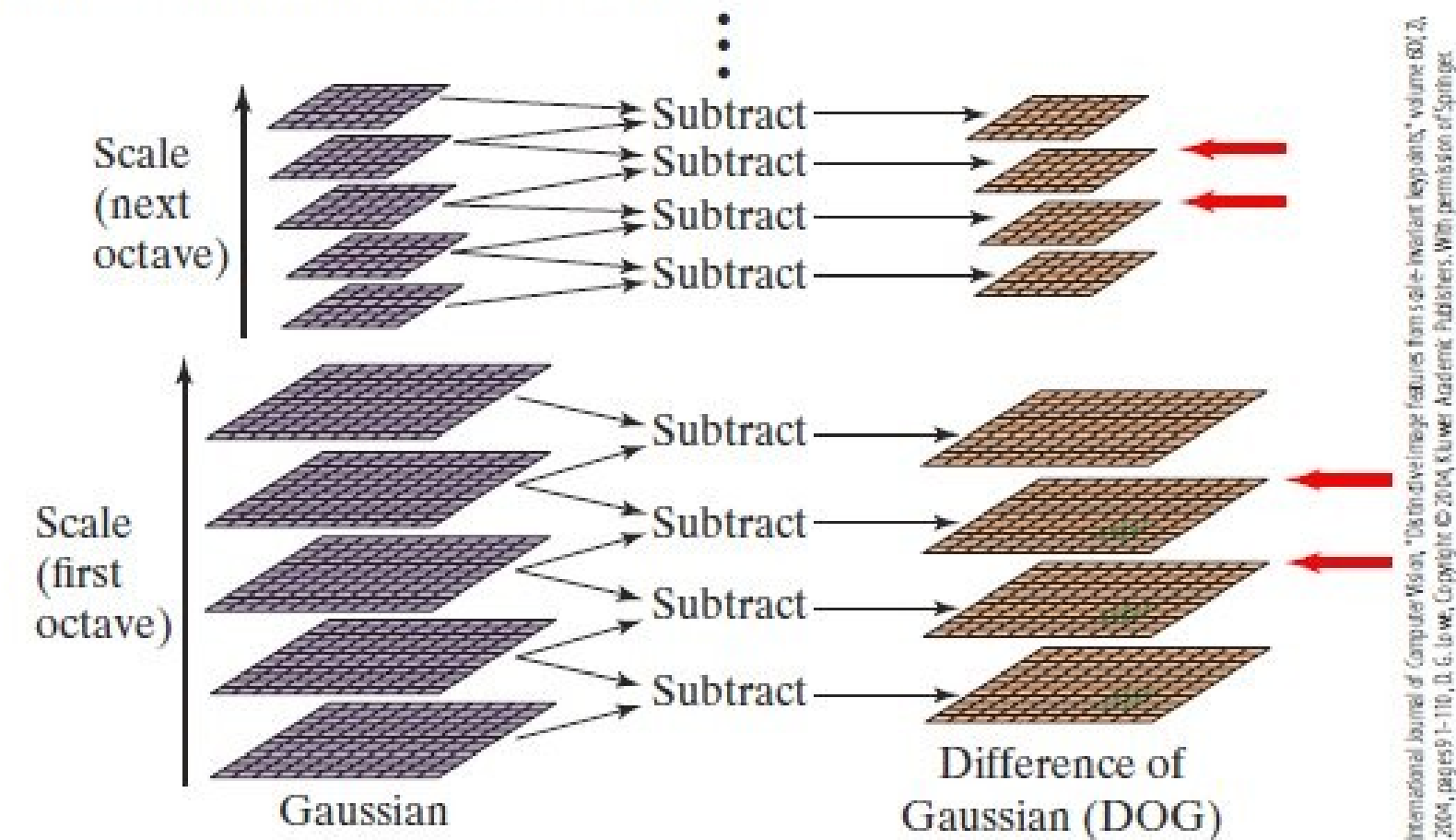
Feature Descriptors



OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

- **Feature detector vs. Feature descriptor:** *What is the difference between their outputs?*
- SIFT Features
- GLOH (Gradient Location & Histogram)
- **HOG** (Histogram of Gradients)
- SURF (Speeded Up Robust Feature) Feature
- ORB (Oriented fast and Robust BRIEF) Feature
- GABOR Texture Features

Figure 7.17 SIFT features are detected by computing a Laplacian pyramid, then looking for local maxima among the 26 neighbors of a pixel. For each octave in this drawing there are 5 Gaussians, 4 LoGs (approximated by DoGs), and two scales (indicated by red arrows); the remaining two LoGs are used only in the neighborhood computation.



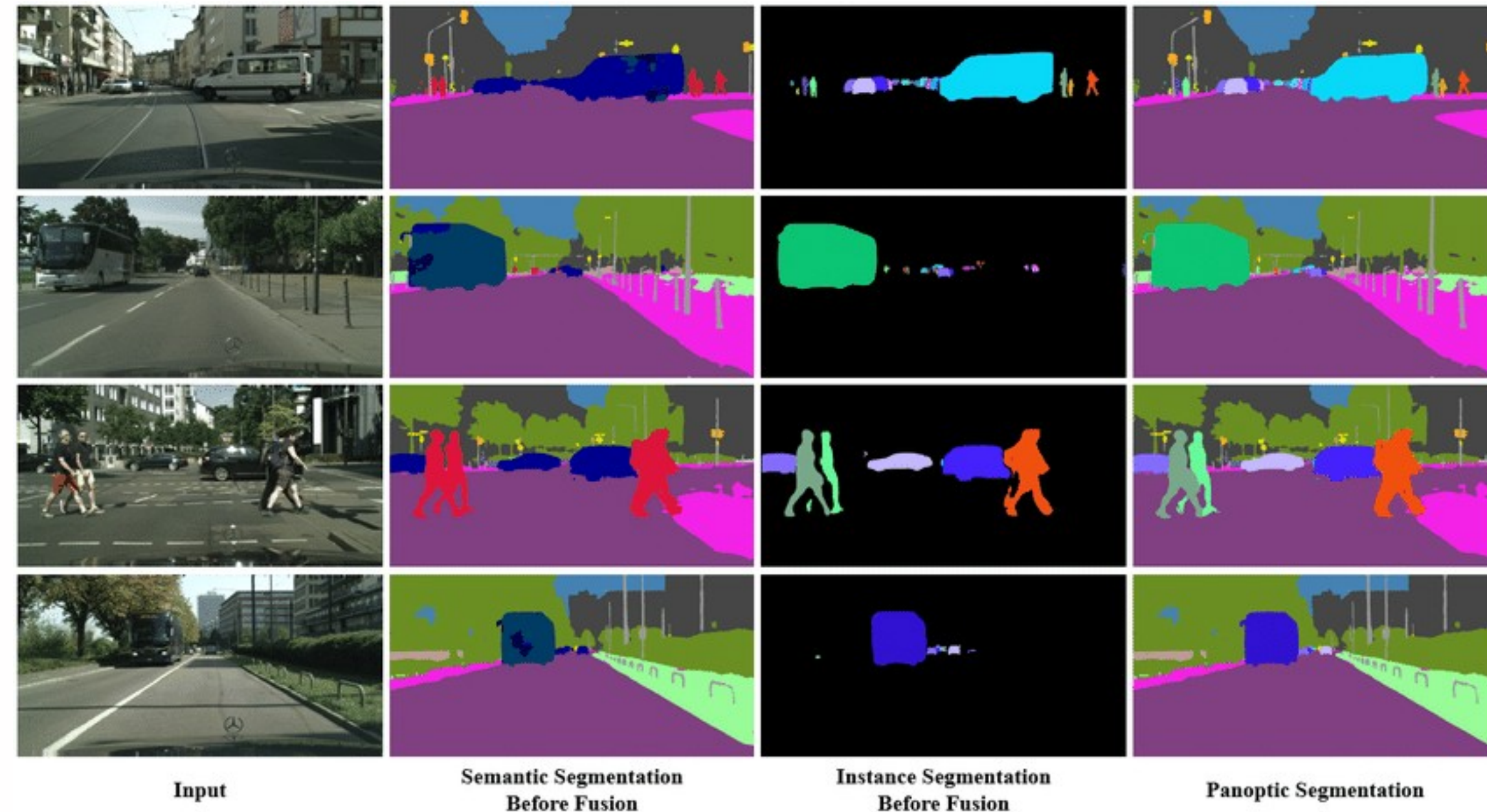
International Journal of Computer Vision, "Data-driven image features from scale-invariant keypoints," volume 63, 2004, pages 91–110, D. G. Lowe, Copyright © 2004, Kluwer Academic Publishers. With permission of Springer.

Image Segmentation



OLLSCOIL NA GAILLIMHÉ
UNIVERSITY OF GALWAY

- A fundamental image **analysis** task
- A ~~bottom-up~~ process that groups pixels in an image based on their low-level properties such as colour or texture.
- Three related tasks defined in recent years:
- ◆ **Semantic Segmentation:** labels every single pixel contained in an image by its semantic class
- ◆ **Instance Segmentation:** focuses only on the semantic classes that can be counted
- ◆ **Panoptic Segmentation:** entails both of the above



Fusion Scheme for Semantic and Instance-level Segmentation – Scientific Figure on ResearchGate.
Available from: https://www.researchgate.net/figure/Panoptic-segmentation-by-unifying-semantic-and-instance-segmentation_fig2_329616112 [accessed 23 Oct 2024]

- ◆ *Have you encountered any segmentation technique so far?*

Image Segmentation



OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

- **Global Thresholding:** Let τ be a single global threshold. Once τ has been determined, it is applied to every pixel in a straightforward manner.

$$I'(x, y) = \begin{cases} \text{ON} & \text{if } I(x, y) > \tau \\ \text{OFF} & \text{otherwise} \end{cases}$$

- Two simple, widely used global thresholding techniques are known as the **Ridler-Calvard** algorithm and **Otsu's method**.

Figure 10.1 LEFT: A grayscale image of several types of objects (fruit) on a dark background (conveyor belt). RIGHT: The graylevel histogram of the image.

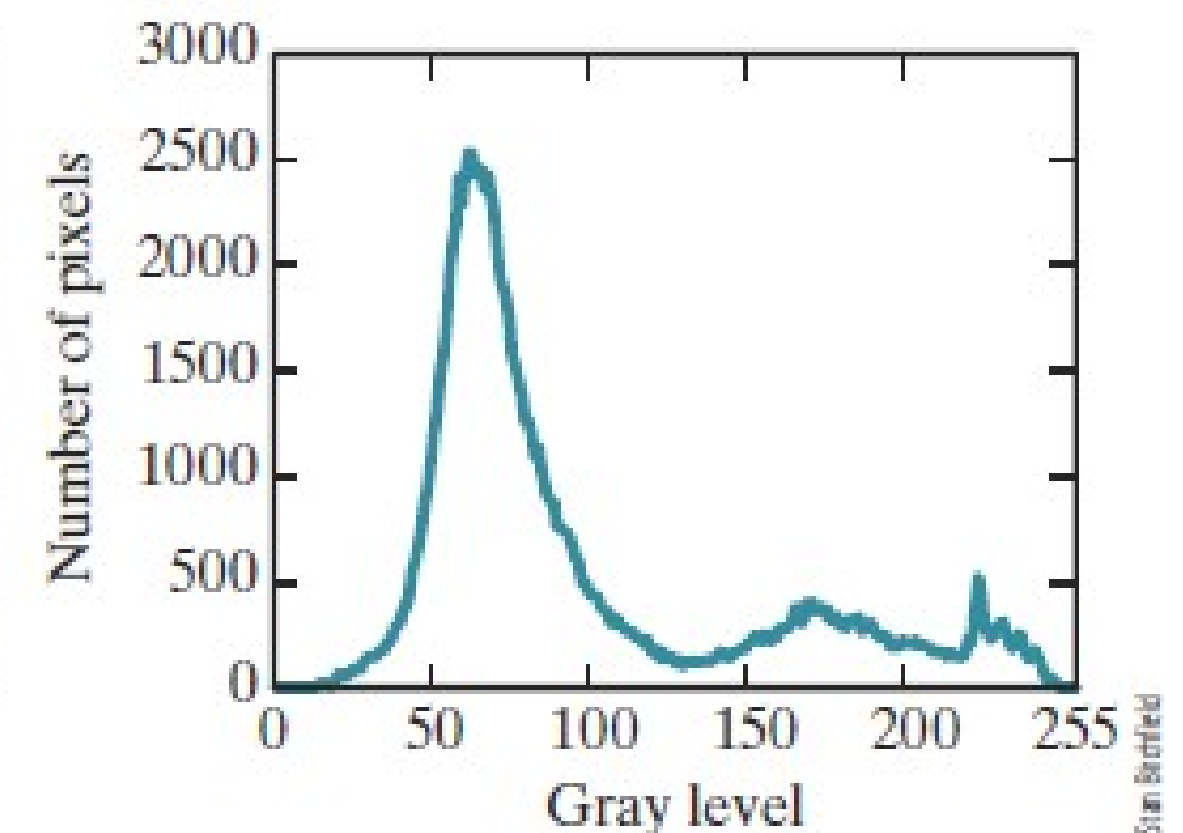
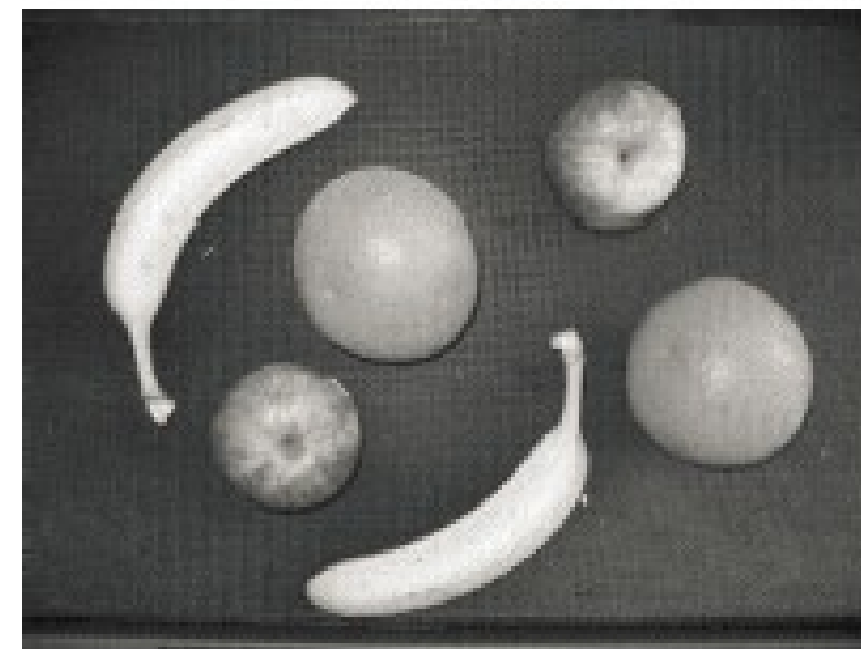


Image Segmentation



- **Global Thresholding Ridler-Calvard Algorithm:**
- Let τ be a threshold, and let μ_{\blacktriangleleft} be the mean gray level of all the pixels whose gray level is less than or equal to τ , while $\mu_{\blacktriangleright}$ is the mean gray level of all the pixels whose gray level is greater than τ . If we assume that the background is darker than the foreground, then μ_{\blacktriangleleft} is the mean of the background pixels, whereas $\mu_{\blacktriangleright}$ is the mean of the foreground pixels.

$$\mu_{\blacktriangleleft} = \frac{m_1[\tau]}{m_0[\tau]} \quad \text{and} \quad \mu_{\blacktriangleright} = \frac{m_1[\zeta - 1] - m_1[\tau]}{m_0[\zeta - 1] - m_0[\tau]} \quad (10.2)$$

$$m_0[\tau] = \sum_{\ell=0}^{\tau} h[\ell] \quad \text{and} \quad m_1[\tau] = \sum_{\ell=0}^{\tau} \ell h[\ell] \quad (10.3)$$

Image Segmentation



- **Global Thresholding Ridler-Calvard Algorithm:**
- Iterate between step 1 and 2 (similar to the 'Expectation-Maximization algorithm')
 - Step 1 (lines 9, 10): Computes the two means based on the current estimate of threshold.
 - Step 2 (line 11): Set the threshold to the average of the two new means.
- Assumes that foreground and background gray levels are distributed as Gaussians with equivalent variance (and standard deviation).

ALGORITHM 10.1 Compute an image threshold using the Ridler-Calvard algorithm

RIDLER-CALVARD(I)

Input: grayscale image I

Output: threshold value τ

```
1   $h \leftarrow \text{COMPUTE\_HISTOGRAM}(I)$ 
2   $m_0[0] \leftarrow h[0]$ 
3   $m_1[0] \leftarrow 0$ 
4  for  $k \leftarrow 1$  to  $\zeta - 1$  do
5       $m_0[k] \leftarrow m_0[k - 1] + h[k]$ 
6       $m_1[k] \leftarrow m_1[k - 1] + k * h[k]$ 
7   $\tau \leftarrow \zeta/2$ 
8  repeat
9       $\mu_{\leftarrow} \leftarrow m_1[\tau]/m_0[\tau]$ 
10      $\mu_{\rightarrow} \leftarrow (m_1[\zeta - 1] - m_1[\tau]) / (m_0[\zeta - 1] - m_0[\tau])$ 
11      $\tau \leftarrow \text{ROUND}(\frac{1}{2}(\mu_{\leftarrow} + \mu_{\rightarrow}))$ 
12 until  $\tau$  does not change
13 return  $\tau$ 
```


Image Segmentation



■ Global Thresholding Ridler-Calvard Algorithm:

Figure 10.2 Step-by-step example of the Ridler-Calvard algorithm applied to the image of Figure 10.1. Note that even with an initial threshold far from the true solution, the algorithm converges in only five iterations. The top row shows the histogram. The green arrow pointing down indicates the threshold at each iteration, while the gold arrows pointing up indicate the two means. The bottom row shows the result of thresholding the image using the threshold for that iteration.

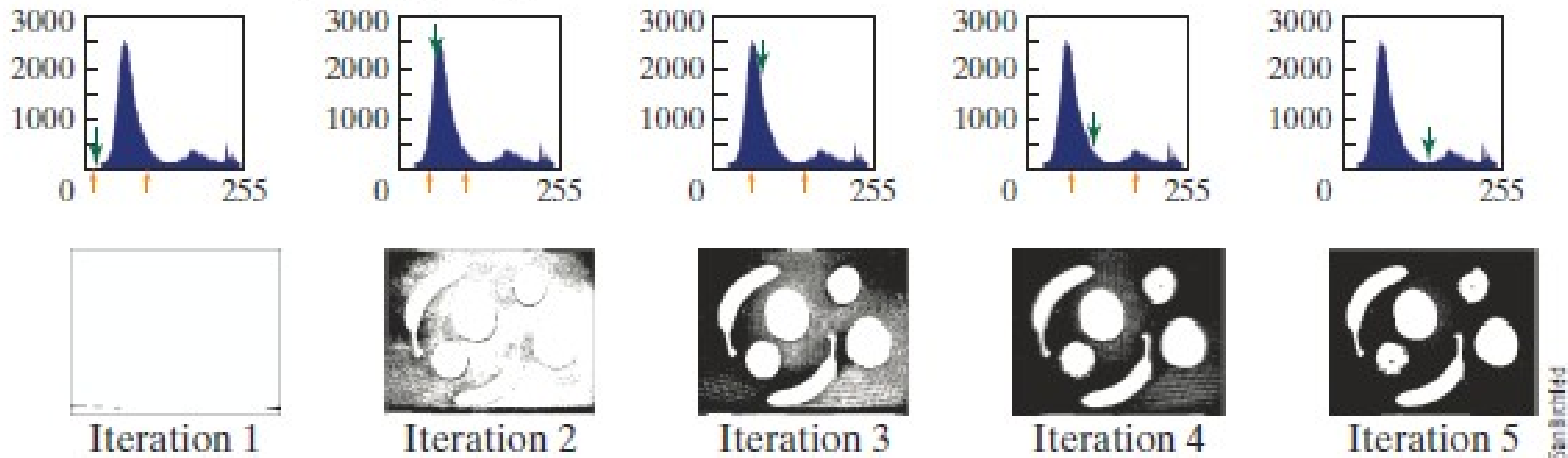


Image Segmentation



■ Global Thresholding: Otsu's Method

Assuming that the underlying distributions can have different variances, find a threshold that minimizes the 'within-class' variance which is defined as the weighted sum of the variances of the two groups of pixels:

$$\sigma_w^2(\tau) \equiv p_{\blacktriangleleft}(\tau)\sigma_{\blacktriangleleft}^2(\tau) + p_{\blacktriangleright}(\tau)\sigma_{\blacktriangleright}^2(\tau) \quad (10.6)$$

■ Compare this with 'between class' variance:

$$\sigma_b^2(\tau) \equiv p_{\blacktriangleleft}(\tau)(\mu_{\blacktriangleleft}(\tau) - \mu)^2 + p_{\blacktriangleright}(\tau)(\mu_{\blacktriangleright}(\tau) - \mu)^2 \quad (10.10)$$

■ Total variance = 'within-class' variance + 'between-class' variance (=

■ Since the total variance does not depend on the threshold, minimizing is the same as maximizing . The advantage of the latter is that it is dependent only upon first-order properties (means) rather than second order properties (variances), thus making it easier to compute

$$\sigma_b^2(\tau) = \frac{(m_1[\tau] - \mu m_0[\tau])^2}{m_0[\tau](m_0[\zeta - 1] - m_0[\tau])} \quad (10.16)$$

Image Segmentation



■ Global Thresholding Otsu's Method

Since there is a small number of possible thresholds (usually just 256), Otsu's method iterates through all these possible values for to find the one that maximizes the quantity

ALGORITHM 10.2 Compute an image threshold using Otsu's method

OTSU(I)

Input: grayscale image I

Output: threshold value τ

```
1   $h \leftarrow \text{COMPUTE\_HISTOGRAM}(I)$ 
2   $m_0[0] \leftarrow h[\ell]$ 
3   $m_1[0] \leftarrow \ell * h[\ell]$ 
4  for  $\ell \leftarrow 1$  to  $\zeta - 1$  do
5       $m_0[\ell] \leftarrow m_0[\ell - 1] + h[\ell]$ 
6       $m_1[\ell] \leftarrow m_1[\ell - 1] + \ell * h[\ell]$ 
7   $\mu \leftarrow m_1[\zeta - 1] / m_0[\zeta - 1]$ 
8   $\hat{\sigma}_b^2 \leftarrow 0$ 
9  for  $\ell \leftarrow 0$  to  $\zeta - 1$  do
10      $\sigma_b^2 \leftarrow (m_1[\ell] - \mu m_0[\ell])^2 / (m_0[\ell] * (m_0[\zeta - 1] - m_0[\ell]))$ 
11     if  $\sigma_b^2 > \hat{\sigma}_b^2$  then
12          $\hat{\sigma}_b^2 \leftarrow \sigma_b^2$ 
13          $\tau \leftarrow \ell$ 
14  return  $\tau$ 
```


Next Time

- Image Segmentation (Part 1 of 2)
- Model Fitting



OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY



OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

Thank *you*